

CLAIMS

Having thus described our invention, what we claim as new and desire to secure by Letters Patent is as follows:

1. A method of multithread processing on a computer, said method comprising:
 - processing a first thread on a first component, said first component capable of simultaneously executing at least two threads;
 - processing said first thread on a second component, said second component capable of simultaneously executing at least two threads; and
 - comparing a result of said processing on said first component with a result of said processing on said second component.
2. The method of claim 1, further comprising:
 - generating a fault signal if said comparison is not equal.
3. The method of claim 1, further comprising:
 - providing an input to enable or to disable said method.
4. The method of claim 1, wherein said processing said thread on said second

YOR920010219US2

component is performed at a priority lower than a priority of said processing said thread on said first component.

5. The method of claim 1, wherein said processing said thread on said second component occurs at a time delayed from that of said processing said thread on said first component.

6. The method of claim 1, wherein said processing said thread on said second component uses information available from said processing said thread on said first component.

7. A method of concurrent fault crosschecking in a computer having a plurality of simultaneous multithreading (SMT) processors, each said SMT processor processing a plurality of threads, said method comprising:

processing a first foreground thread and a first background thread on a first SMT processor; and

processing a second foreground thread and a second background thread on a second SMT processor,

wherein said first background thread executes a check on said second foreground thread and said second background thread executes a check on said

YOR920010219US2

first foreground thread, thereby achieving a crosschecking of said first SMT processor and said second SMT processor.

8. The method of claim 7, wherein said first foreground thread has a higher priority than that of said first background thread and said second foreground thread has a higher priority than that of said second background thread.

9. The method of claim 7, further comprising:

storing each of a result of said processing said first foreground thread and said processing said second foreground thread in a memory for subsequent comparison with a corresponding result of said first and second background threads.

10. The method of claim 7, further comprising:

communicating, between said first SMT processor and said second SMT processor, a thread branch outcome for said first foreground thread and for said second foreground thread.

11. The method of claim 7, further comprising:

generating a signal if either of said checks are unequal.

YOR920010219US2

12. The method of claim 7, further comprising:

providing a signal to enable or disable said concurrent fault crosschecking.

13. A computer, comprising:

a first simultaneous multithreading (SMT) processor; and

a second simultaneous multithreading (SMT) processor,

wherein said first SMT processor processes a first foreground thread and a first background thread and said second SMT processor processes a second foreground thread and a second background thread, and

wherein said first background thread executes a check on said second foreground thread and said second background thread executes a check on said first foreground thread.

14. The computer of claim 13, wherein said first foreground thread has a higher priority than that of said first background thread, and said second foreground thread has a higher priority than that of said second background thread.

15. The computer of claim 13, further comprising:

a delay buffer storing a result of said first foreground thread; and

a delay buffer storing a result of said second foreground thread.

YOR920010219US2

16. The computer of claim 13, further comprising:

a memory storing a result of a thread branch outcome for said first foreground thread and a result of a thread branch outcome for said second foreground thread.

17. The computer of claim 13, further comprising:

a logic circuit comparing a result of said first foreground thread with a result of said second background thread and generating a signal if said results are not equal; and

a logic circuit comparing a result of said second foreground thread with a result of said first background thread and generating a signal if said results are not equal.

18. The computer of claim 13, further comprising:

an input signal to determine whether said crosschecking process is one of enabled and disabled.

19. The computer of claim 13, further comprising:

a memory storing an information related to said processing by each of said first and second foreground threads, thereby providing to the respective first and second background threads an information to expedite processing.

YOR920010219US2

20. The computer of claim 13, further comprising:

at least one output signal signifying that a result of at least one of said first and second background threads does not agree with a respective result of a check of said first and second foreground threads.

21. The computer of claim 13, comprising a plurality of pairs of SMT processors,

wherein each said pair comprises a first simultaneous multithreading (SMT) processor and a second simultaneous multithreading (SMT) processor, said first SMT processor processes a first foreground thread and a first background thread and said second SMT processor processes a second foreground thread and a second background thread, and

said first background thread executes a check on said second foreground thread and said second background thread executes a check on said first foreground thread.

22. The computer of claim 16, wherein said memory storing said results of a thread branch outcome comprises a first memory for said first foreground thread and a second memory for said second foreground thread.

YOR920010219US2

23. A multiprocessor system executing a method of multithread processing on a computer, said method comprising:

processing a first thread on a first component, said first component capable of simultaneously executing at least two threads;

processing said first thread on a second component, said second component capable of simultaneously executing at least two threads; and

comparing a result of said processing on said first component with a result of said processing on said second component.

24. An Application Specific Integrated Circuit (ASIC) containing a medium tangibly embodying a program of machine-readable instructions executable by a digital processing apparatus to perform a method of multithread processing, said method comprising:

processing a first thread on a first component, said first component capable of simultaneously executing at least two threads;

processing said first thread on a second component, said second component capable of simultaneously executing at least two threads; and

comparing a result of said processing on said first component with a result of said processing on said second component.

YOR920010219US2

25. A Read Only Memory (ROM) containing a medium tangibly embodying a program of machine-readable instructions executable by a digital processing apparatus to perform a method of multithread processing, said method comprising:

processing a first thread on a first component, said first component capable of simultaneously executing at least two threads;

processing said first thread on a second component, said second component capable of simultaneously executing at least two threads; and

comparing a result of said processing on said first component with a result of said processing on said second component.